

Tema 3: Jerarquía de Memorias

• Conceptos Avanzados de Memoria Principal

Departament d'Arquitectura de Computadors

Facultat d'Informàtica de Barcelona

Universitat Politècnica de Catalunya

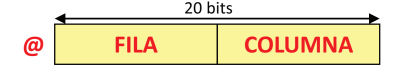


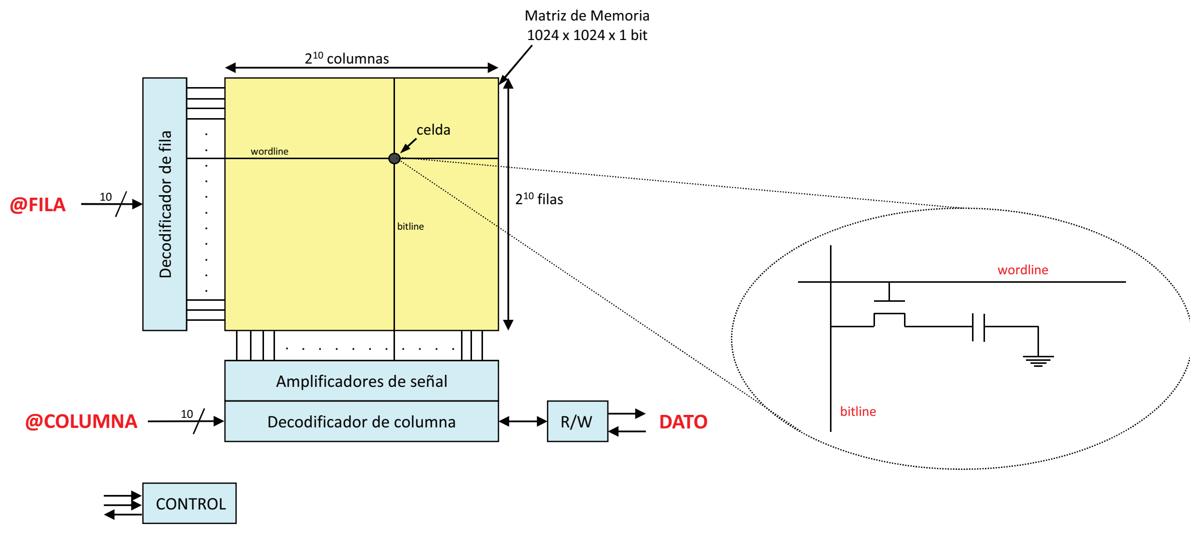


Índice

- Memoria Principal
- Conceptos Básicos Memoria Cache
- Memoria Virtual
- Conceptos Avanzados Memoria Cache
- Conceptos Avanzados Memoria Principal
  - Introducción
  - Estructura interna de una DRAM actual
  - Accesos múltiples
  - Organización de la Memoria Principal
  - Refresco en una DRAM
  - Detección y Corrección de Errores
- Juntándolo todo: Visión General de la Jerarquía

Estructura básica de una DRAM



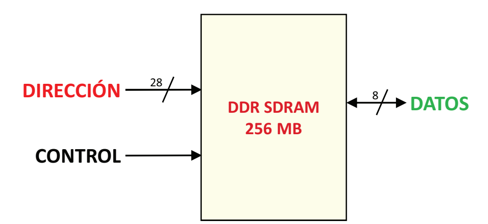




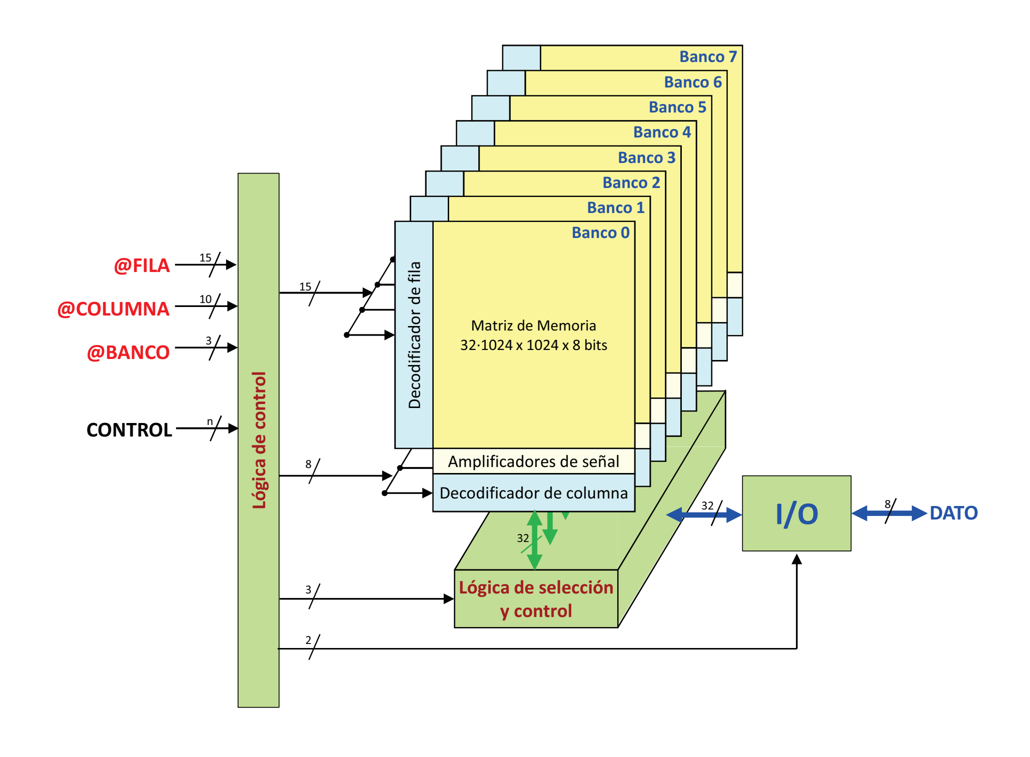
Estructura actual de una DRAM

- Una característica fundamental de las DRAM actuales es que están divididas en **bancos**.
- **Los bancos permiten:**
  - Realizar accesos concurrentes a diferentes bancos
  - Ocultar la precarga
  - Ocultar el refresco
  - Tener arrays de memoria más pequeños:
    - ✓ Tiempo de acceso ↓
    - ✓ Consumo de energía↓

- **Por ejemplo, una DDR3 de 256 MB tiene:**
  - 8 bancos de 32 MB
  - Cada banco tiene 32 K filas por 1K columnas de 1 byte.



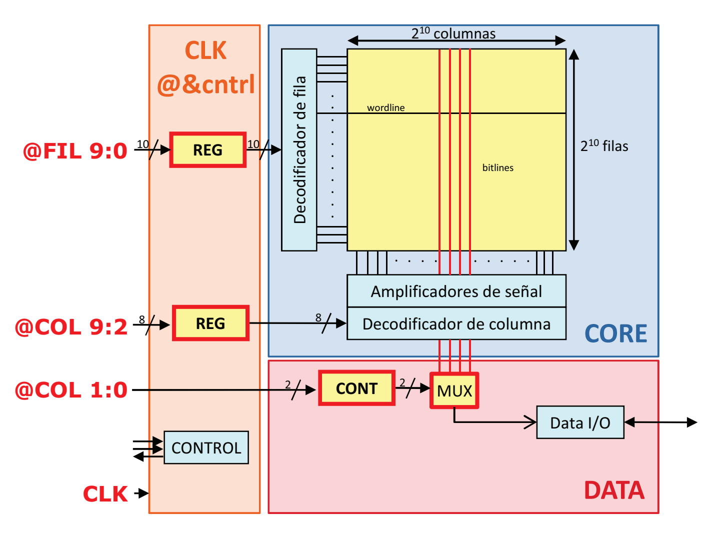
Estructura actual de una DRAM





Variantes de SDRAM

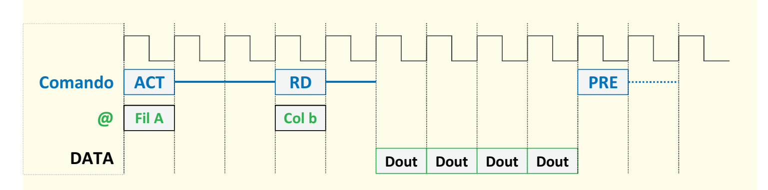
- **SDR (Single Data Rate), DDR (Double Data Rate), DDR2, DDR3**



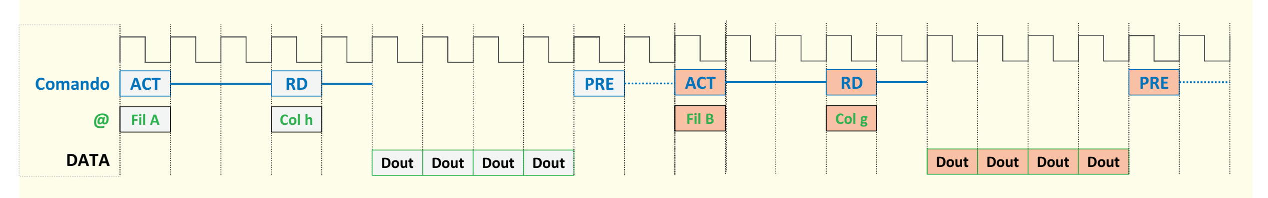
Estándar y ejemplos	CORE Frecuencia (MHz)	CLK Frecuencia (MHz)	DATA M datos / segundo
SDR	1x	1x	1x
SDRAM-133	133	133	133
SDRAM-166	166	166	166
SDRAM-200	200	200	200
DDR	1x	1x	2x
DDR266	133	133	266
DDR300	150	150	300
DDR400	200	200	400
DDR2	1/2x	1x	2x
DDR2-533	133	266	533
DDR2-667	166	333	667
DDR2-800	200	400	800
DDR3	1/4x	1x	2x
DDR3-1066	133	533	1066
DDR3-1333	166	667	1333
DDR3-1600	200	800	1666

Accesos a una DRAM con 1 banco

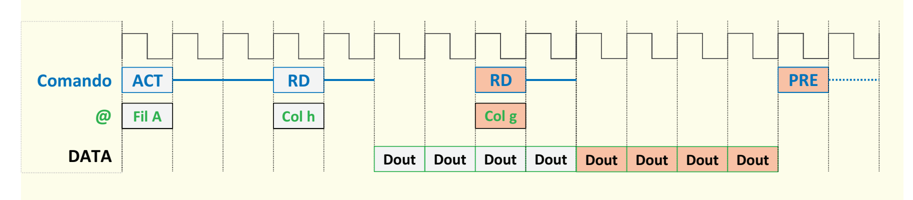
- **Lectura de una línea (Fila A, columna b)**



- **Lectura de 2 líneas en Filas diferentes (Fila A, columna h y Fila B, columna g)**



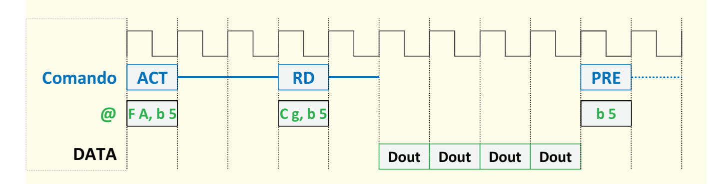
- **Lectura de 2 líneas en la misma página (Fila A, columna h y Fila A, columna g)**



- 32 bytes por línea
- ACT: 3 ciclos
- WR, RD: 2 ciclos
- PRE: 2 ciclos
- Transferencia: 8B por ciclo

Accesos a una DRAM con múltiples bancos

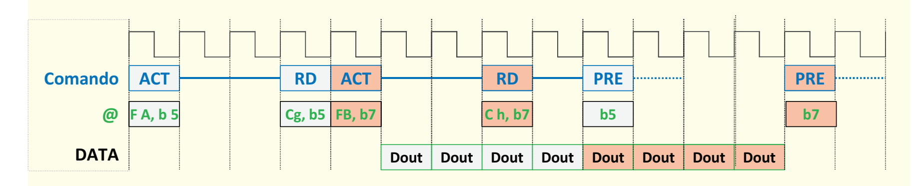
- **Lectura de una línea (Banco 5, Fila A, Columna g)**



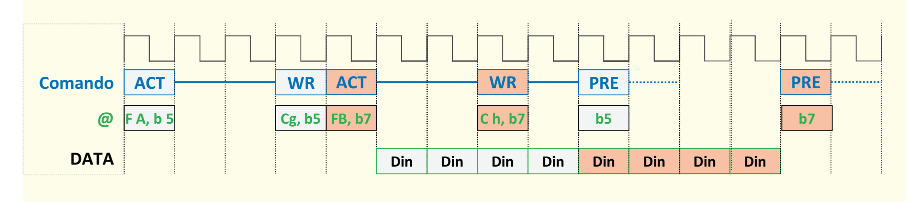
En una DRAM real existen bastantes restricciones que no hemos tenido en cuenta, p.e.:

- Entre una lectura y una escritura hay que esperar N ciclos.
- Entre 2 ACT consecutivos a bancos diferentes han de pasar M de ciclos.

- **Lectura de 2 líneas en bancos diferentes (Banco 5, Fila A, Columna g y Banco 7, Fila B, columna g)**



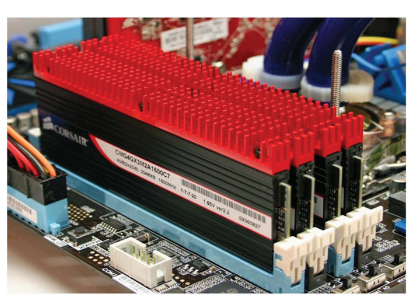
- **Escritura de 2 líneas en bancos diferentes (Banco 5, Fila A, Columna g y Banco 7, Fila B, columna g)**



- 32 bytes por línea
- ACT: 3 ciclos
- WR, RD: 2 ciclos
- PRE: 2 ciclos
- Transferencia: 8B por ciclo

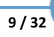
Organización de la Memoria Principal

- **¿Qué tenemos aquí?**
  - 8GB de memoria DDR3 1600 MHz
  - Direcciones de 33 bits

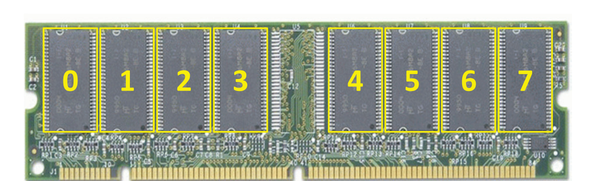


www.blog.corsair.com

- **¿Cómo está organizado?**
  - Canales (2)
  - DIMMs (2 por canal)
  - Chips (8 por DIMM)
  - Bancos (8 por chip)
  - Filas (32·1024 por Banco)
  - Columnas (1024 por Fila)
  - 1 byte por columna
- **¿Dónde esta la dirección 0x1DF038A6B?**
- **Dada una dirección, ¿cómo identificamos canal, DIMM, chip, …?**

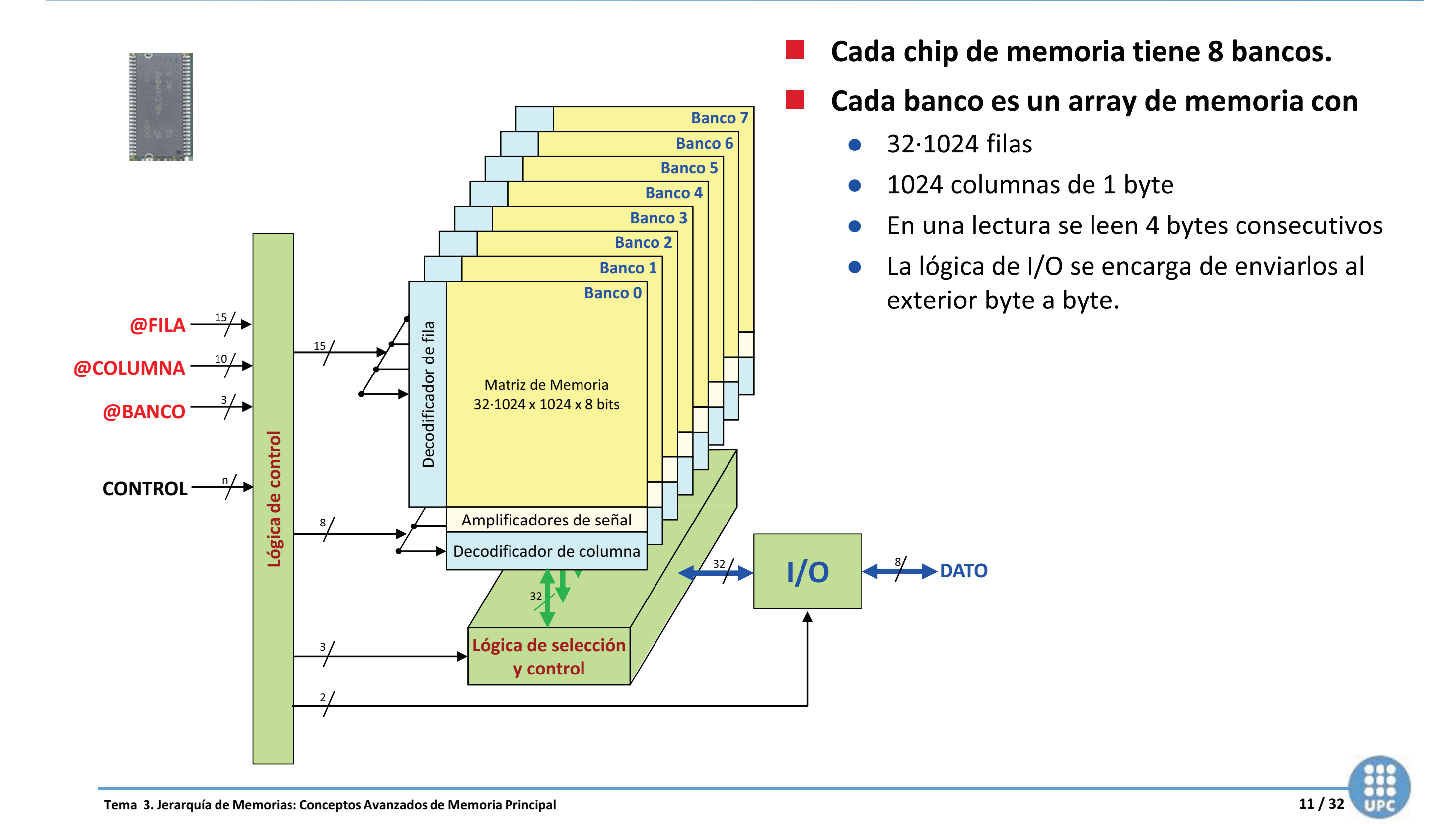



Organización de la Memoria Principal





- **La Memoria Principal utiliza DIMMS** (*Dual Inline Memory Modules*).
- **A nivel de ejemplo, un DIMM podría tener las siguientes características:**
  - 2GB de capacidad (requiere 31 bits de dirección)
  - Permite leer 64 bits (8 bytes) de datos en un acceso individual
  - 8 chips
- **Cada DIMM de memoria tiene un número determinado de chips.**
- **A nivel de ejemplo, un chip podría tener las siguientes características:**
  - 256 MB de capacidad (requiere 28 bits de dirección)
  - Permite leer 8 bits (1 byte) de datos en un acceso individual
  - 8 bancos



¿Dónde esta la dirección 0x1DF038A6B?

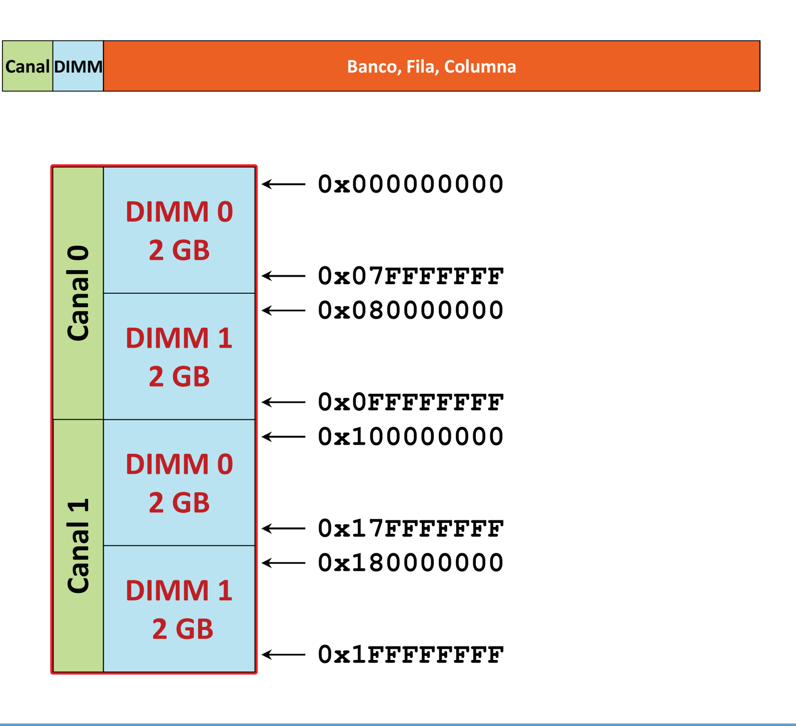
- **Nuestra memoria tiene una capacidad de 8GB (33 bits de dirección)**
- **En nuestra Memoria tenemos:**
  - Canales (2): 1 bit para el canal
  - DIMMs (2 por canal): 1 bit para el DIMM
  - Chips (8 por DIMM): 3 bits para el chip
  - Bancos (8 por chip): 3 bits para el banco
  - Filas (32·1024 por Banco): 15 bits para la fila
  - Columnas (1024 por Fila): 10 bits para la columna
- **Existen muchas posibilidades válidas:**

Canal	DIMM	Banco	Fila	Columna	Chip	
Canal	DIMM	Fila	Banco	Columna	Chip	
Canal	Fila	DIMM	Banco	Columna	Chip	
Canal	DIMM	Fila	Columna$_H$	Banco	Columna$_L$	Chip

¡ El rendimiento del Sistema de Memoria está muy influenciado por esta decisión !

¿Dónde esta la dirección 0x1DF038A6B?

- El mapa de memoria viene determinado por esta descomposición:



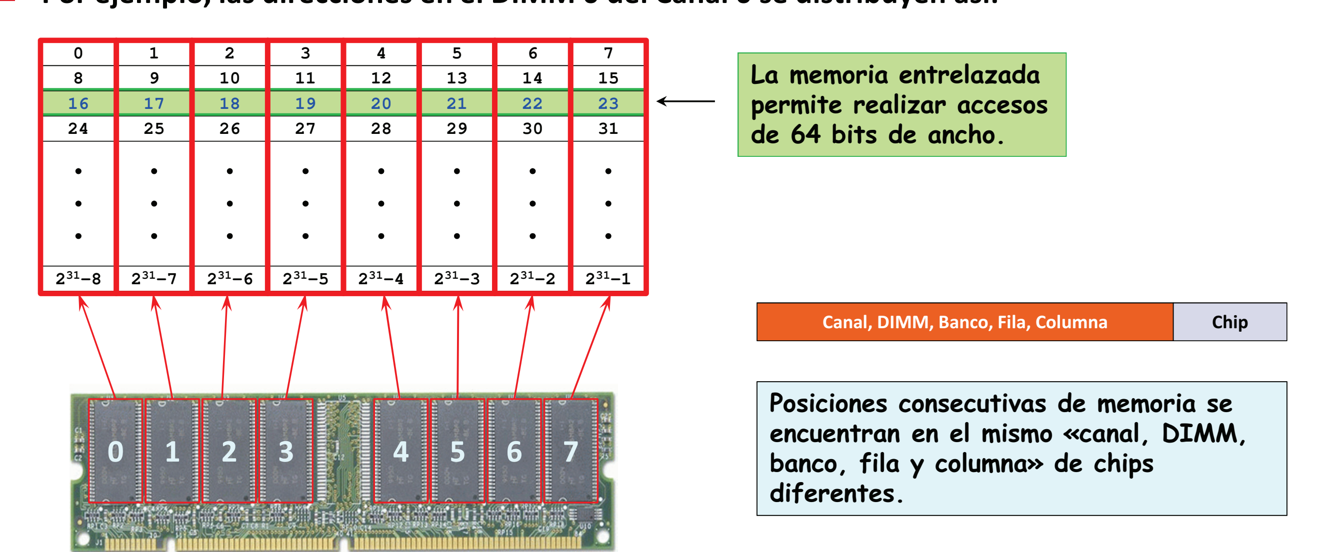


¿Dónde esta la dirección 0x1DF038A6B?

- Es **imprescindible** que los bits bajos de la dirección seleccionen el chip dentro del DIMM

→ MEMORIA ENTRELAZADA

- Por ejemplo, las direcciones en el DIMM 0 del Canal 0 se distribuyen así:



¿Dónde esta la dirección 0x1DF038A6B?

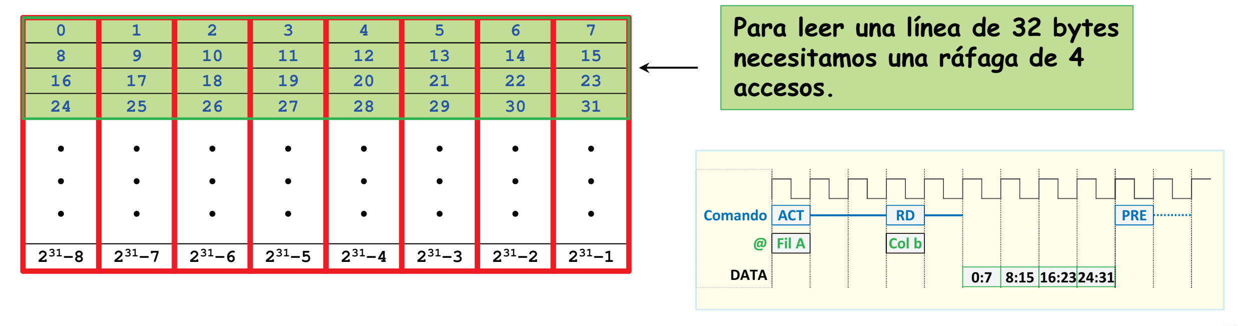
- Todos los accesos a Memoria Principal leen (o escriben) líneas de cache.

# línea	byte en la línea

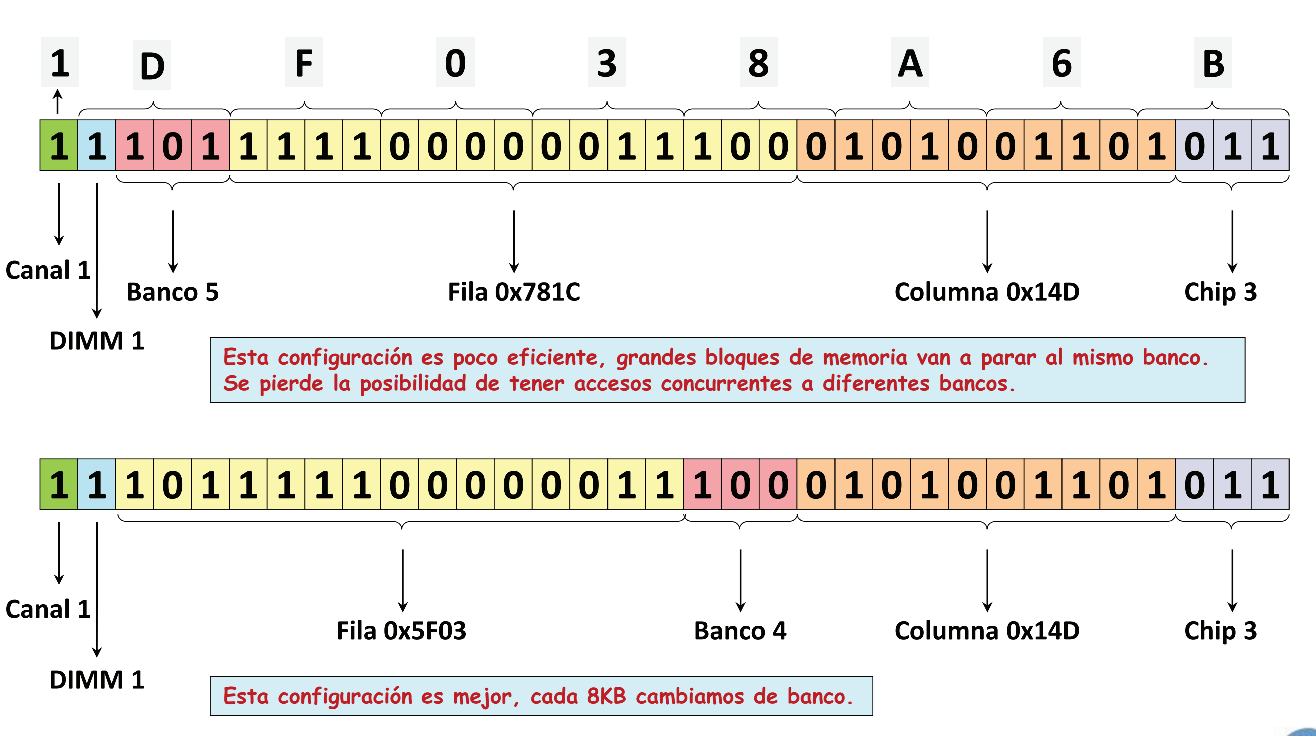
- Las DRAM actuales permiten leer (o escribir) de forma muy eficiente ráfagas de datos.
- Para que esas ráfagas coincidan con líneas de cache es imprescindible que el campo correspondiente al byte en la línea corresponda a los bits que seleccionan chip y columna$_L$.

#línea = (Canal, DIMM, Banco, Fila, Columna$_H$)	Col$_L$	Chip

- Por ejemplo, con una línea de 32 bytes, la línea 0 se encuentra en el canal 0, DIMM 0:



¿Dónde esta la dirección 0x1DF038A6B?



¿Dónde esta la dirección 0x1DF038A6B?

1	1	1	0	1	1	1	1	1	0	0	0	0	0	0	1	1	1	0	0	0	1	0	1	0	0	1	1	0	1	0	1	1

Canal 1
DIMM 1
Banco 5
Fila 0x781C
Columna 0x14D
Chip 3

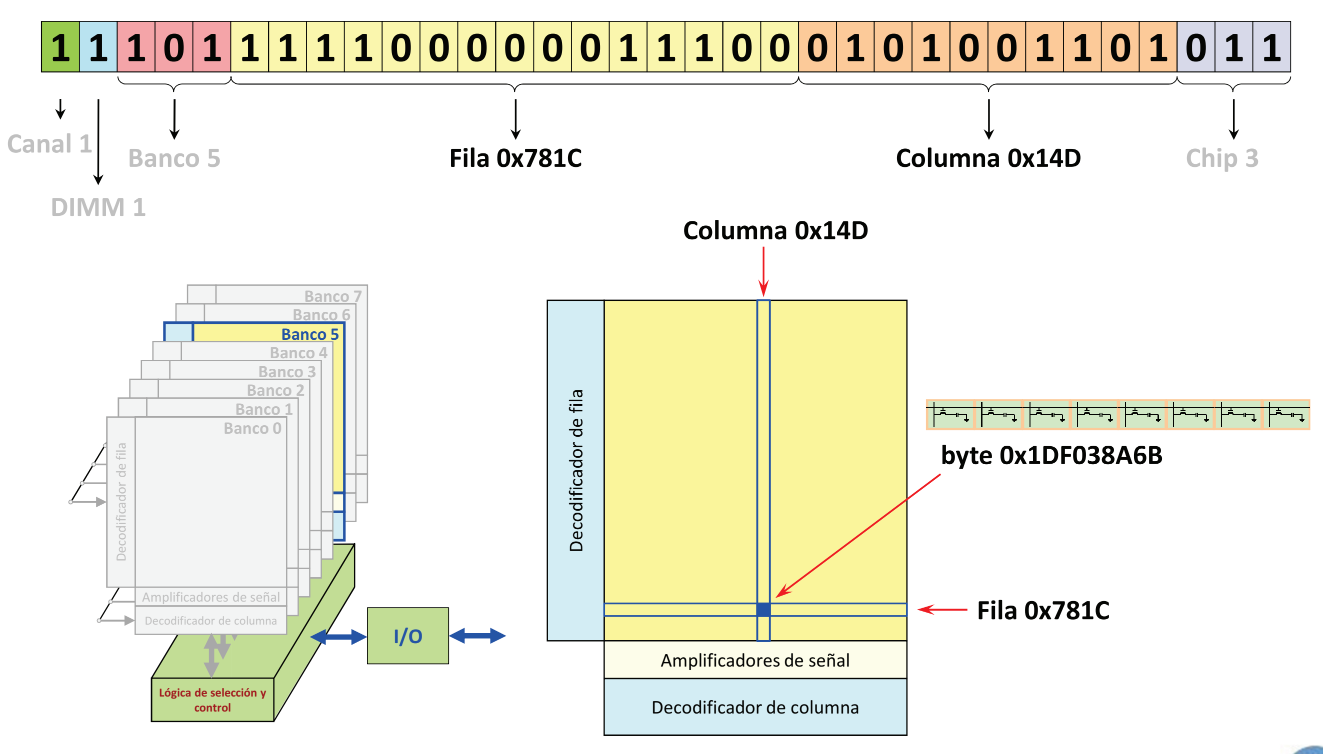
	DIMM 0 2 GB
Canal 0	DIMM 1 2 GB
Canal 1	DIMM 0 2 GB
	DIMM 1 2 GB

0 1 2 3 4 5 6 7

Banco 7
Banco 6
Banco 5
Banco 4
Banco 3
Banco 2
Banco 1
Banco 0
Decodificador de fila
Amplificadores de señal
Decodificador de columna
Lógica de selección y control
I/O

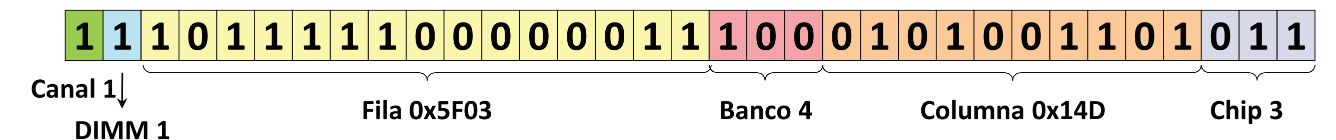
Esta configuración sólo es un ejemplo de implementación.

¿Dónde esta la dirección 0x1DF038A6B?



Lectura línea

- La transmisión de datos entre MP y MC se hace por líneas completas
- Ejemplo: 32 bytes por línea, acceso provocado por un fallo en la dirección 0x1DF038A6B



- Hay que leer todos los bytes de la línea:

1	1	1	0	1	1	1	1	1	0	0	0	0	0	0	1	1	1	0	0	0	1	0	1	0	0	1	1	-	-	-	-	-

- Si transferimos la línea en orden hay que iniciar el acceso con la siguiente dirección:

1	1	1	0	1	1	1	1	1	0	0	0	0	0	0	1	1	1	0	0	0	1	0	1	0	0	1	1	0	0	-	-	-

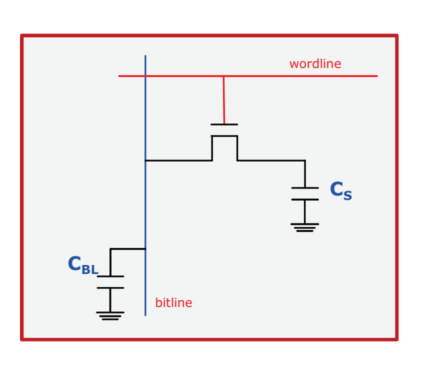
- Si transferimos en primer lugar el dato que ha provocado en fallo, la dirección es:

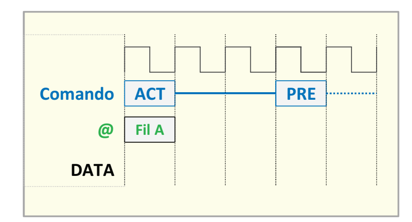
1	1	1	0	1	1	1	1	1	0	0	0	0	0	0	1	1	1	0	0	0	1	0	1	0	0	1	1	0	1	-	-	-

Problema del refresco en una DRAM

- **Cada celda de una DRAM equivale a un condensador, como tal, va perdiendo carga de forma exponencial:**
  - Corriente de fuga.
  - La lectura es destructiva.
- **Para evitar la perdida de información almacenada en las celdas, la carga debe regenerarse periódicamente**

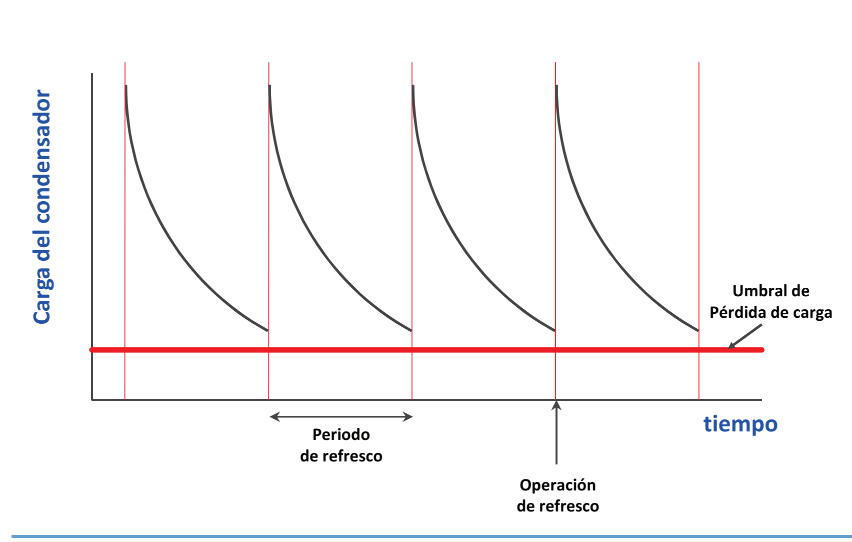
  → REFRESCO
- **El refresco hay que aplicarlo a todas las celdas de la memoria. Se aplica fila a fila.**
- **El refresco de la fila A se puede hacer con un ACT y una PRE.**
- **Existen comandos especiales para hacer el refresco.**
  - Refrescan todos los bancos a la vez.
  - La propia memoria controla que fila hay que refrescar.





Problema del refresco en una DRAM

- **Actualmente el refresco es una operación estandarizada establecida por el JEDEC (Junction Electronic Devices Engineering Council)**
  - Periodo de refresco de 64ms.
  - Si tenemos 32·1024 filas, *en media*, hay que lanzar una operación de refresco cada 2µs.



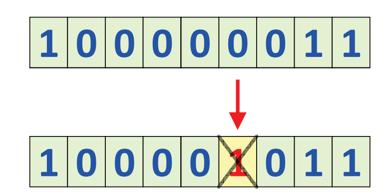


- Frecuencia: 1 GHz
- Coste refrescar fila: 20 ciclos
- Cada 64ms se pierden 655360 ciclos.
- Se pierden $10{,}2 \cdot 10^6$ ciclos por segundo.
- Si el refresco NO se puede solapar, se pierde el 1% del tiempo refrescando la memoria.

¡El coste de refrescar la memoria no es despreciable!

Detección y Corrección de Errores

- **Al acceder a memoria se pueden producir errores:**



Dato almacenado en Memoria

Dato que le llega a la CPU

- **Estos errores se deben a diversas causas:**
  - Interferencias electrónicas o magnéticas (**transitorios**)
  - Problemas de transmisión de datos (**transitorios**)
  - Problemas hardware (**permanentes**), como en cualquier otro chip y por las mismas causas.

En el IBM Watson Research Center se midió para una muestra de DRAMs clónicas una tasa de error de 5950 fallos cada 10^9 horas de funcionamiento a nivel del mar (1 fallo cada 19 años, 2 meses, 7 días y 19 horas) . La misma prueba en una mina, bajo una capa de rocas de 15 metros, no produjo ningún error. Estos errores son debidos a los rayos cósmicos.

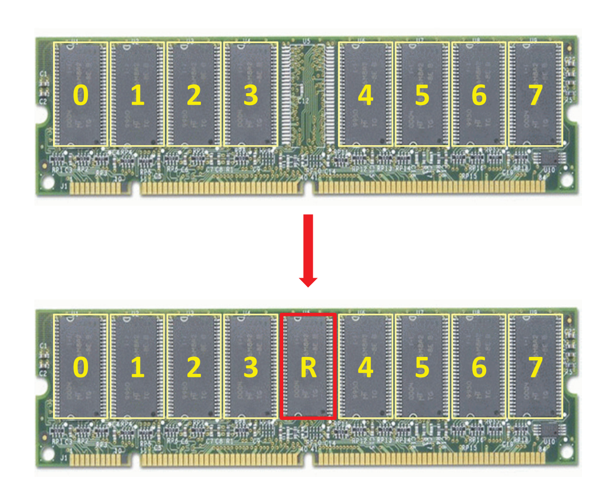


En una instalación como el BSC con alrededor de 8.000 procesadores y contando un mínimo de 8 chips de memoria por procesador se produce un fallo cada 2 horas y 37 minutos (8 por día, 240 al mes, 2880 al año, …)

Detección y Corrección de Errores

- En algunas instalaciones los errores han de estar bajo control.
- Los errores permanentes sólo se resuelven sustituyendo el elemento dañado.
- Los fabricantes de memoria diseñan sistemas que permiten detectar y en algunos casos corregir los errores transitorios.

→ AÑADIR INFORMACIÓN REDUNDANTE

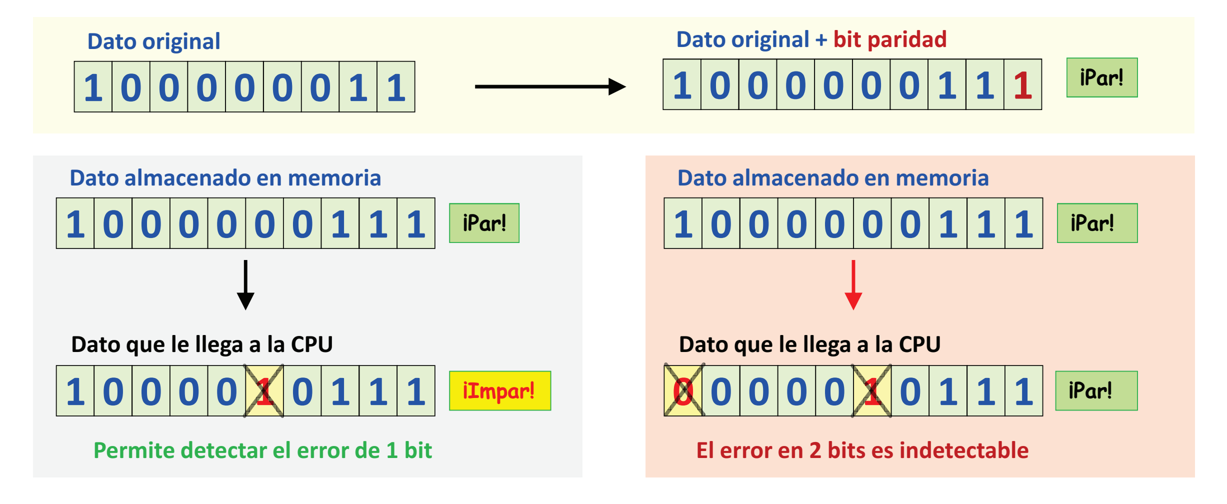




Detección y Corrección de Errores

Bit de Paridad

- El sistema más simple es añadir 1 bit, de tal forma que cada porción de información tenga SIEMPRE un número impar (o par) de 1's.



Este sistema sólo sirve para detectar un número impar de errores.
No permite corregir errores.

Detección y Corrección de Errores

Código Hamming (7, 3). 4 bits de datos y 3 redundantes

- Bits redundantes: 1, 2, 4
- Bits de datos: 3, 5, 6 y 7

7 111	6 110	5 101	4 100	3 011	2 010	1 001	
b3	b2	b1	p2	b0	p1	p0	Palabra de 7 bits
111 b3	-	101 b1	-	011 b0	-	001 p0	p0 = b3 xor b1 xor b0
111 b3	110 b2	-	-	011 b0	010 p1	-	p1 = b3 xor b2 xor b0
111 b3	110 b2	101 b1	100 p2	-	-	-	p2 = b3 xor b2 xor b1

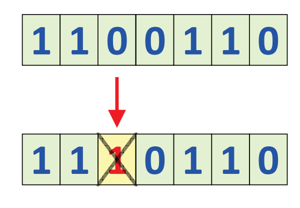
7	6	5	4	3	2	1	
1	1	0	p2	1	p1	p0	Palabra de 7 bits
1	-	0	-	1	-	0	p0 = 1 xor 0 xor 1 = 0
1	1	-	-	1	1	-	p1 = 1 xor 1 xor 1 = 1
1	1	0	0	-	-	-	p2 = 1 xor 1 xor 0 = 0

Posibles implementaciones		
Bits Paridad	Bits Datos	total
3	4	7
4	11	15
5	26	31
6	57	63
7	120	127
m	2^m-m-1	2^m-1

Detección y Corrección de Errores

Código Hamming (7, 3). 4 bits de datos y 3 redundantes

- Permite **corregir** un error en 1 bit.



Error en el bit 5

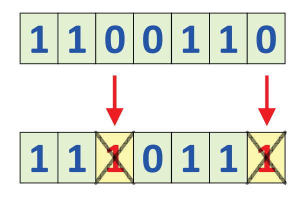
b3 7	b2 6	b1 5	p2 4	b0 3	p1 2	p0 1	Test Hamming
1	1	1	0	1	1	0	
1	-	1	-	1	-	0	T0 = p0 xor b3 xor b1 xor b0 = 0^1^1^1 = 1
1	1	-	-	1	1	-	T1 = p1 xor b3 xor b2 xor b0 = 1^1^1^1= 0
1	1	1	0	-	-	-	T2 = p2 xor b3 xor b2 xor b1 = 0^1^1^1= 1

El test Hamming da T2T1T0=101, indicando que el error está en el bit 5.
Si T2T1T0=000 no ha habido ningún error.

Detección y Corrección de Errores

Código Hamming extendido (7, 3) + 1 bit de paridad.

- El código Hamming **no funciona** cuando se produce un **error en 2 bits**.



Error en los bits 1 y 5

b3 7	b2 6	b1 5	p2 4	b0 3	p1 2	p0 1	Test Hamming
1	1	1	0	1	1	1	
1	-	1	-	1	-	1	T0 = p0 xor b3 xor b1 xor b0 = 1^1^1^1 = 0
1	1	-	-	1	1	-	T1 = p1 xor b3 xor b2 xor b0 = 1^1^1^1= 0
1	1	1	0	-	-	-	T2 = p2 xor b3 xor b2 xor b1 = 0^1^1^1= 1

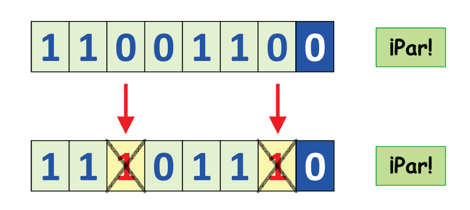
El test Hamming da T2T1T0=001, indicando que hay un error en el bit 1.
No hay información del bit 5.

- **Añadiendo un bit de paridad es posible detectar un error en 2 bits o corregir un error de 1 bit.**



Detección y Corrección de Errores

Código Hamming extendido (7, 3) + 1 bit de paridad.

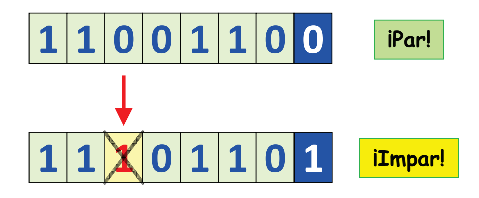


¡Par!

¡Par!

Error en los bits 1 y 5

No hay error de paridad.
El test Hamming da T2T1T0=001
Esto implica que se ha producido un error en 2 bits.
No se puede recuperar.



¡Par!

¡Impar!

Error en el bit 5

Error de paridad.
El test Hamming da T2T1T0=101
Esto implica que se ha producido un error en el bit 5.
Se puede recuperar.

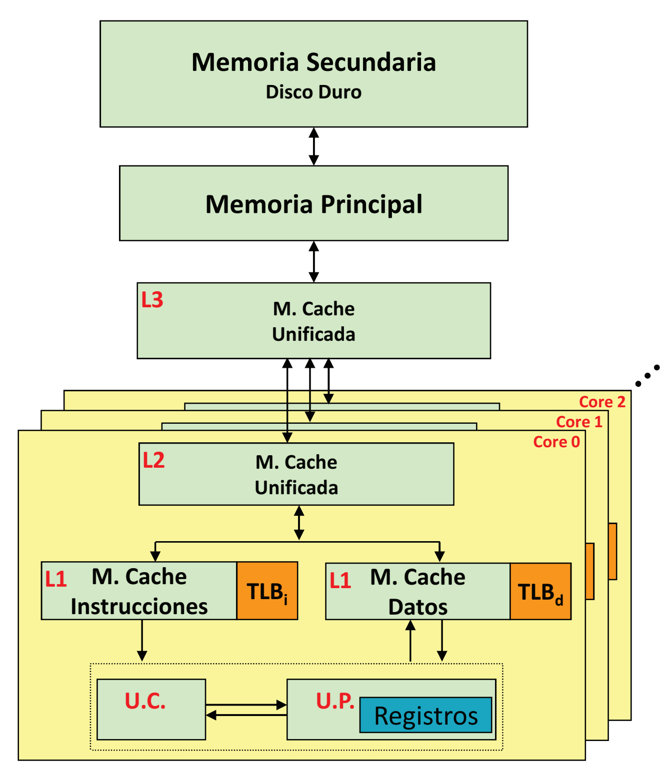
Detección y Corrección de Errores

Memoria ECC (Error Correction Code)

- La memoria que incluye estos mecanismos se utiliza exclusivamente en servidores y estaciones de trabajo de gama alta.
- Basado en **códigos de Hamming**. A cada porción de datos se añade información redundante que permite detectar y corregir errores.
- El código ECC más utilizado en memorias es el **SEC-DED** (Single Error Correction – Double Error Detection). Permite corregir 1 error o detectar 2 en la misma porción de datos.
- En DIMMs de memoria se suele utilizar un código extendido (71, 64) + bit de paridad. 72 bits totales, con 64 bits de datos y 8 redundantes. Perfecto para utilizar en un DIMM con 9 chips de 8 bits de ancho.

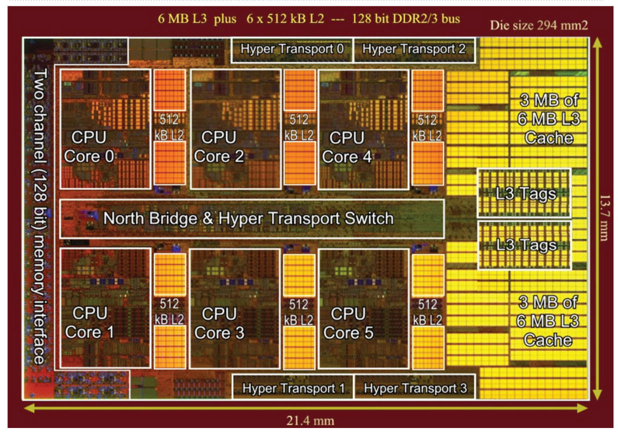


Juntándolo todo



Ejecución de instrucción **incl (%ebx)**

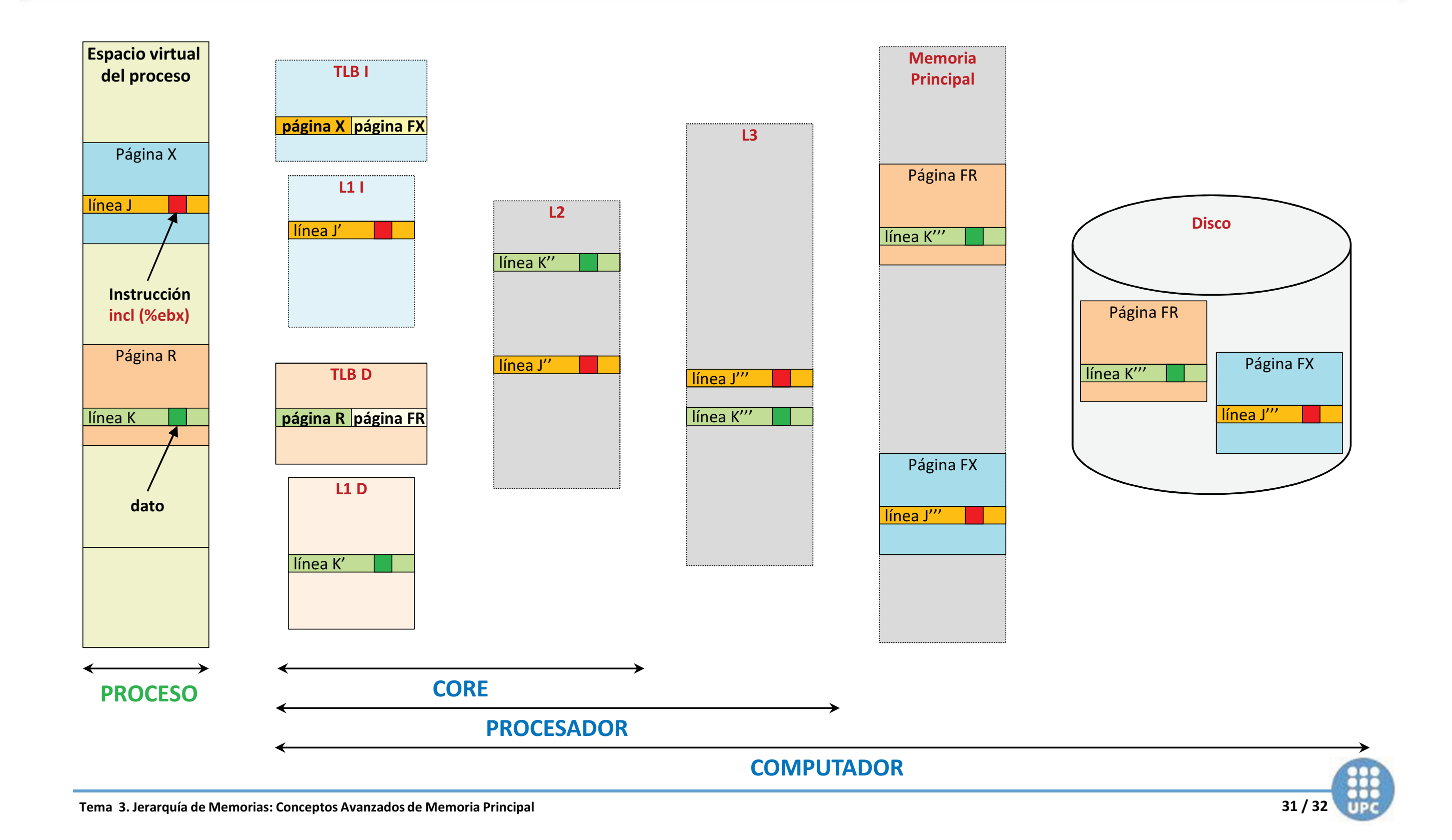
En el **PEOR CASO** puede provocar:
- Al leer instrucción: fallo TLBi, fallo página (acceso a disco), fallo L1 instrucciones, fallo L2, fallo L3
- Al leer dato: fallo TLBd, fallo página (acceso a disco), fallo L1 datos, fallo L2, fallo L3
- Al escribir dato: acierto TLBd, acierto L1 datos



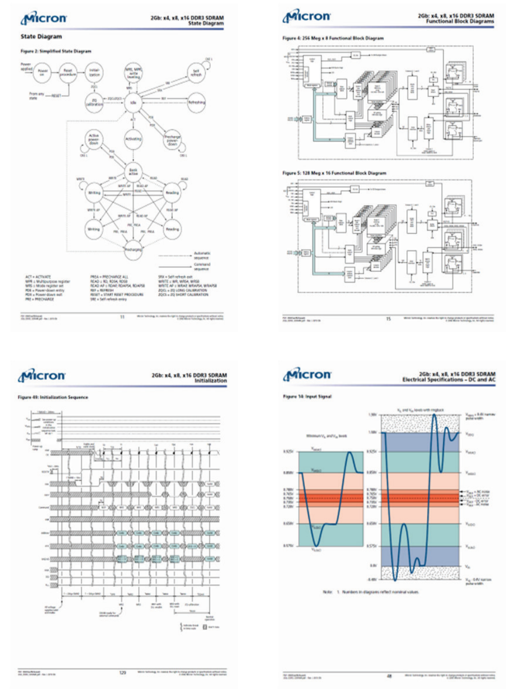
The Istanbul Opteron die. Source: AMD



Juntándolo todo



El mundo real



Imágenes obtenidas en www.micron.com

- **Detalles de la DDR3 SDRAM MT41J512M4 de Micron.**
- **Así durante 207 páginas.**
- **La vida real es mucho más compleja.**